UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/976,726	10/12/2001	Jason King	5150-58700	2464

35690 7590 09/11/2006

MEYERTONS, HOOD, KIVLIN, KOWERT & GOETZEL, P.C.
700 LAVACA, SUITE 800
AUSTIN, TX 78701

EXAMINER	
ZHOU, TING	

ART UNIT	PAPER NUMBER
2173	

DATE MAILED: 09/11/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 09/976,726
Filing Date: October 12, 2001
Appellant(s): KING ET AL.

Mark S. Williams
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 15 June 2006 appealing from the Office action mailed 6 January 2006.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is substantially correct. The changes are as follows:

Claims 80-81 stand rejected under 35 U.S.C 112, first paragraph, as failing to comply with the written description requirement.

WITHDRAWN REJECTIONS

The following grounds of rejection are not presented for review on appeal because they have been withdrawn by the examiner:

The 35 U.S.C. 112 first paragraph rejection for claims 16, 71 and 87 has been withdrawn in view of the applicant's amendments filed on 11 April 2006.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

5,651,108	CAIN ET AL.	7-1997
6,578,174	ZIZZO	6-2003

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claims 80 and 81 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the written description requirement. The claim(s) contains subject matter which was not described

in the specification in such a way as to reasonably convey to one skilled in the relevant art that the inventor(s), at the time the application was filed, had possession of the claimed invention. The limitations "wherein said receiving the third user input explicitly specifying the one or more user interface events to configure for the first node *does not* include receiving user input specifying a connection between the first node and a second node" and "wherein said receiving the third user input explicitly specifying the one or more user interface events to configure for the first node is performed *independently* of configuring other nodes in the block diagram of the graphical program", of claims 80 and 81 respectively, are not positively recited in the specification of the present application. "Any negative limitation or exclusionary proviso must have basis in the original disclosure. If alternative elements are positively recited in the specification, they may be explicitly excluded in the claims....The mere absence of a positive recitation is not basis for an exclusion" (MPEP 2173.05 (i)). The specification does not explicitly recite the exclusion of receiving user input specifying a connection between the first node and a second node and configuring other nodes in the block diagram of the graphical program. Therefore, there is no positively recited basis for the negative limitations of claims 80 and 81.

Claims 1-7, 10-25, 32-44, 46-60, 62-79 and 82-89 are rejected under 35 U.S.C. 102(b) as being anticipated by Cain et al. U.S. Patent 5,651,108 (hereinafter "Cain").

Referring to claims 1, 19 and 32, Cain teaches a method and memory medium comprising creating a graphical user interface for the graphical program in response to first user input (for example, users opening the graphical user interface shown in Figure 4A for providing event-

based, interactive visual-programming) (column 3, lines 18-45 and column 7, lines 18-32); displaying a first node for receiving user interface events in a block diagram for the graphical program in response to second user input (displaying a button in response to user selection for receiving methods which will execute in response to programmed user events) (column 3, lines 18-65, column 7, line 63-column 8, line 14 and column 10, line 50-column 12, line 46); receiving third user input explicitly specifying one or more user interface events to configure for the first node (users can change or attach new user interface events which the button will respond to via input on the graphical user interface, as described with relation to Figures 4D-4E) (column 11, line 30-column 12, line 46); configuring the first node to receive the one or more user interface events explicitly specified by the third user input during execution of the graphical program (during execution of the program, i.e. in the run mode, the button object will respond to user defined interface events such as generation of a push-button event upon clicking on the button) (column 11, line 30-column 12, line 46); and associating one or more portions of graphical code with the first node in response to fourth user input, wherein each portion of graphical code comprises one or more nodes for responding to one or more of the user interface events which the first node is configured to receive (attaching program code to a button such that the code executes when the button is clicked) (column 10, line 48-column 11, line 3).

Referring to claims 23 and 86, Cain teaches a method comprising creating a graphical user interface for the graphical program in response to user input (for example, users opening the graphical user interface shown in Figure 4A for providing event-based, interactive visual-programming) (column 3, lines 18-45 and column 7, lines 18-32); creating a block diagram for the graphical program in response to user input, wherein creating the block diagram comprises

creating a first portion of graphical code in response to user input, wherein the first portion of graphical code includes one or more nodes for responding to user interface events (creating a block diagram such as the diagram shown in Figure 5F for example, via a plurality of selected objects such as buttons for receiving methods which will execute in response to programmed user events) (column 3, lines 18-65, column 7, line 63-column 8, line 14 and column 10, line 50-column 12, line 46); receiving user input explicitly specifying one or more user interface events to associate with the first portion of graphical code (attaching program code specifying a response to user interface events such as users clicking the button is attached to the button) (column 3, lines 18-65 and page 10, line 50-column 11, line 29); and configuring the first portion of graphical code to execute in response to the one or more explicitly specified user interface events being generated during execution of the graphical program (upon the specified user interface event being generate, i.e. user clicking on the button, the program code attached to the button is executed to perform a function such as displaying a "Hello. World!" message in a dialog box) (column 3, lines 18-65 and page 10, line 50-column 11, line 29).

Referring to claims 36, 53 and 66, Cain teaches a method, memory medium and system comprising memory for storing program instructions (Figure 1A); a processor coupled to the memory (Figure 1A); and a display device (Figure 1A); wherein the processor is operable to execute program instructions stored in the memory to: display a first node in a block diagram of the graphical program in response to user input (displaying a button in response to user selection for receiving methods which will execute in response to programmed user events) (column 3, lines 18-65, column 7, line 63-column 8, line 14 and column 10, line 50-column 12, line 46); associate a first portion of graphical source code with the first node in response to user input

(attaching program code to buttons) (column 10, lines 50-58), wherein the first portion of graphical source code associated with the first node comprises a plurality of interconnected nodes (nodes can be interconnected with, i.e. contained within other nodes, as shown by the data diagram of interconnected nodes shown in Figure 5G) (column 14, line 56-column 17, line 24); associate a first user interface event with the first node in response to user input explicitly specifying the first user interface event (users can attach or associate user interface events which the button will respond to via input on the graphical user interface, as described with relation to Figures 4D-4E) (column 11, line 30-column 12, line 46); and configure the first portion of graphical source code associated with the first node to execute in response to the first user interface event associated with the first node being generated during execution of the graphical program (program code can be attached to buttons so that the code executes in response to specified user interface events such as users clicking the button) (column 3, lines 18-65 and column 10, line 50-column 12, line 46).

Referring to claims 2, 20 and 33, Cain teaches wherein the first node comprises one or more sub-diagrams (objects contained within other objects), wherein associating the one or more portions of graphical code with the first node comprises displaying each portion of graphical code within one of the sub-diagrams of the first node (objects contained with other objects inherits the events from the parent object) (column 14, line 56-column 17, line 24).

Referring to claims 3, 21, 34, 72-73, Cain teaches wherein the receiving the user input explicitly specifying the one or more user interface events to configure for the first node comprises receiving user input explicitly specifying one or more user interface events to which each of the sub-diagrams of the first node corresponds; wherein for each portion of graphical

code, displaying the portion of graphical code within one of the sub-diagrams of the first node comprises configuring the portion of graphical code to execute in response to the one or more user interface events to which the sub-diagram corresponds (objects can be contained within other objects and each object displayed on the GUI can have user interface events specified by the user, which in turn as has attached program code that executes in response to an event such as user selection) (column 10, line 48-column 11, line 3 and column 14, line 56-column 17, line 24).

Referring to claims 4 and 35, Cain teaches wherein for each portion of graphical code, displaying a portion of graphical code within one of the sub-diagrams of the first node comprises displaying the one or more nodes of the portion of graphical code within one of the sub-diagrams of the first node (objects can be contained within other objects and each object has attached program code that executes in response to an event such as user selection) (column 10, line 48-column 11, line 3 and column 14, line 56-column 17, line 24).

Referring to claims 5, 51 and 64, Cain teaches wherein the block diagram comprises a data flow block diagram (as shown by the "Object Tree" in Figure 5D for example).

Referring to claims 6, 25, 37, 49, 50 and 54, Cain teaches executing the graphical program (executing the program via the run mode) (column 11, lines 17-29); wherein one or more user interface events which the first node is configured to receive are generated during execution of the graphical program, wherein the generating the first user interface comprises generating the first user interface event in response to user input to the graphical user interface (users can change or attach new user interface events which the button will respond to via input on the graphical user interface, as described with relation to Figures 4D-4E) (column 11, line 30-

column 12, line 46); wherein the method further comprises executing one of the portions of graphical code associated with the first node in response to each of the one or more user interface events which the first node is configured to receive being generated during execution of the graphical program (during execution of the program, i.e. in the run mode, the button object will respond to user defined interface events such as generation of a push-button event upon clicking on the button; attaching program code to a button such that the code executes at run-time when the button is clicked) (column 10, line 48-column 11, line 3 and column 11, line 30-column 12, line 46).

Referring to claim 7, Cain teaches wherein the one or more user interface events generated during execution of the graphical program are generated in response to user input to the graphical user interface of the graphical program (such as user clicking a button) (column 10, line 48-column 11, line 3 and column 11, line 30-column 12, line 46).

Referring to claim 10, Cain teaches wherein the configuring the first node to receive the one or more user interface events comprises configuring the first node to receive notification when the one or more user interface events are generated during execution of the graphical program (the node, i.e. the button is notified of user event signals such as mouse button clicks and drags in order to execute the associated method in response to the user event) (column 3, lines 24-56).

Referring to claim 11, Cain teaches wherein the configuring the first node to receive the one or more user interface events comprises configuring the first node to receive information specifying occurrences of the one or more user interface events during execution of the graphical program (the node, i.e. the button is notified of when an user interface event such as mouse

button clicks and drags has occurred in order to execute the associated method in response to the user event) (column 3, lines 24-56).

Referring to claims 12, 40, 57 and 70, Cain teaches displaying the first portion of graphical source code within the first node in response to user input, wherein the first portion of graphical source code associated with the first node is visibly displayed within the first node of the block diagram of the graphical program (for example, Figures 4E and 4F) (column 10, line 48- column 13, line 17).

Referring to claims 13, 46, 48 and 62-63, Cain teaches wherein the configuring the first node to receive the one or more user interface events comprises configuring the first node to receive a first user interface event; wherein the first user interface event explicitly specifies a first user interface element of the graphical user interface and an action performed on the first user interface element (configuring the node, i.e. the button to display a "Hell. World!" message upon an event such as user selection) (column 10, line 65-column 11, line 29).

Referring to claims 14 and 47, Cain teaches wherein the first interface element comprises one of an indicator, a control, a menu element and a window (the button is programmed to display a dialog box window upon user clicking the button) (column 10, line 65-column 11, line 3).

Referring to claims 15, 44 and 60, Cain teaches displaying a first graphical user interface dialog for configuring the first node (Figures 4E and 4D); wherein the receiving the third user input explicitly specifying the one or more user interface events to configure for the first node comprises receiving user input to the first graphical user interface dialog to explicitly specify the

one or more user interface events (specifying user interface events allows users to define and attach method to buttons) (column 11, line 30-column 12, line 46).

Referring to claims 16 and 71, as best understood by the examiner, Cain teaches displaying a second node for dynamically registering user interface events in the block diagram in response to user input (a plurality of nodes or objects such as buttons that can be created by the user) (column 14, line 30-column 17, line 24 and Figure 5D); wherein the receiving the third user input explicitly specifying the one or more user interface events to configure for the first node comprises receiving user input explicitly specifying a first user interface event to dynamically register during execution of the graphical program (Figure 4E shows the receiving of user input specifying methods corresponding to user interface events to be executed during the run-mode) (column 10, line 50-column 12, line 46), wherein the method further comprises configuring the second node to dynamically register the first user interface event during execution of the graphical program (Figure 4E shows the receiving of user input specifying methods corresponding to user interface events to be executed during the run-mode, such as the first user interface event of clicking the button; furthermore, each node, or button can have a plurality of associated events such as clicking the button) (column 10, line 50-column 12, line 63 and Figures 6E and 6F); wherein after dynamically registering the first user interface event, the first node is operable to receive the first user interface event (program code corresponding to how a node, i.e. button respond to specified user interface events are attached to the button so that in the run mode, the button is operable to receive user interface events such as clicking the button and respond accordingly) (column 10, line 50-column 12, line 46).

Referring to claim 17, Cain teaches wherein the configuring the second node dynamically register the first user interface event during execution of the graphical program comprises connecting the second node to the first node in response to user input (user selection of objects and buttons creates the interconnected nodes shown in the "Object Tree" of Figure 5D) (column 16, line 31-column 17, line 67).

Referring to claim 18, Cain teaches wherein the one or more user interface events specified by the third user input includes a first user interface event (specifying events such as the events shown in Figure 4E) (column 3, lines 24-56 and column 10, line 50-column 12, line 49); wherein the method further comprises displaying a second node dynamically un-registering user interface events in the block diagram in response to user input; configuring the second node to dynamically un-register the first user interface event during execution of the graphical program (as shown in Figure 4E, events can be un-registered, i.e. deleted so that they are not part of the program code for the button during run time); wherein before the dynamically un-registering the first user interface event, the first node is operable to receive the first user interface event; wherein after the dynamically un-registering the first user interface event, the first node is not operable to receive the first user interface event (users can receive events listed in the built-in method box shown in Figure 4E before unregistering, i.e. deleting, an event, at which time the event will no longer be a part of, i.e. responded to by the button) (column 11, line 4-column 12, line 49).

Referring to claim 22, Cain teaches wherein the one or more programmatic events comprises one or more of a user interface event, a system event, a timer event, an event

generated in response to data acquired from a device (programming buttons to respond to user interface events such as user clicking a button) (column 10, line 48-column 11, line 3).

Referring to claims 24, 38-39, 52, 55-56 and 65, Cain teaches wherein the first portion of graphical code includes a plurality of nodes (a plurality of interconnected nodes, as shown in the "Object Tree" of Figure 5D), wherein the creating the first portion of graphical code in response to user input comprises arranging the plurality of nodes on a display and interconnecting the plurality of nodes in response to user input (user selection of objects and buttons creates the interconnected nodes shown in the "Object Tree" of Figure 5D) (column 16, line 31-column 17, line 67), wherein the plurality of interconnected nodes visually indicates functionality for responding to the one or more user interface events associated with the first portion of graphical code (the plurality of interconnected nodes visually indicate functionality such as containership relationships which controls how user interface events are handled) (column 14, line 56-column 17, line 67).

Referring to claims 41, 58, 78 and 83, Cain teaches displaying information explicitly identifying a plurality of user interface events, wherein the receiving the user input explicitly specifying the one or more user interface events to associate with the first portion of graphical code comprises receiving user input selecting the one or more user interface events from the displayed information (users can select and custom create user interface events from the displayed events shown in window 461 of Figure 4E) (column 11, line 55-column 12, line 13).

Referring to claims 42 and 59, Cain teaches associating a second portion of graphical source code with the first node in response to user input (each node, or button, can have a plurality of default and custom methods with associated program code that execute in response to

events) (column 11, line 55-column 12, line 63 and Figures 6E and 6F); associating a second user interface event with the first node in response to user input explicitly specifying the second user interface event and configuring the second portion of graphical source code associated with the first node to execute in response to the second user interface event associated with the first node (each node, or button can have a plurality of associated events such that a plurality of program codes corresponding to the plurality of user-defined methods are executed in response to the plurality of events) (column 11, line 55-column 12, line 63 and Figures 6E and 6F).

Referring to claims 43, 68 and 85, Cain teaches wherein the receiving the user input explicitly specifying the one or more user interface events to associate with the first portion of graphical code comprises receiving user input specifying names of the one or more user interface events to associate with the first portion of graphical code (users can create new user interface events, i.e. name new events to be associated with a button via window 461 of Figure 4E; users can also specify and change the names of the buttons) (column 11, line 4-column 12, line 46).

Referring to claims 67 and 84, Cain teaches displaying a list of user interface events, wherein the receiving the user input explicitly specifying the one or more user interface events to associate with the first portion of graphical code comprises receiving user input selecting the one or more user interface events from the displayed list of user interface events (users can select and custom create user interface events from the displayed list of events shown in window 461 of Figure 4E) (column 11, line 55-column 12, line 13).

Referring to claim 69, Cain teaches wherein the associating the one or more portions of graphical code with the first node comprises associating a first portion of graphical code and a second portion of graphical code with the first node (each node, or button can have a plurality of

associated events such that a plurality of program codes corresponding to the plurality of user-defined methods are executed in response to the plurality of events) (column 11, line 55-column 12, line 63 and Figures 6E and 6F); wherein the first node comprises a plurality of sub-diagrams (objects contained within other objects) ("Object Tree" shown in Figure 5D); wherein associating the first portion of graphical code with the first node comprises displaying the first portion of graphical code within a first sub-diagram of the first node; and wherein associating the second portion of graphical code with the first node comprises displaying the second portion of graphical code within a second sub-diagram of the first node (objects contained with other objects inherits the events, i.e. program code associated with events, from the parent object) (column 14, line 56-column 17, line 24).

Referring to claim 74, Cain teaches wherein each of the portions of graphical code is located separately from the first node (for example, as shown in Figure 5B, each of the objects in the object tree, and therefore, their associated portion of program code are an individual and separate entity).

Referring to claim 75, Cain teaches wherein for each of the portions of graphical code, the associating the portion of graphical code with the first node comprises associating the portion of graphical code with one or more of the user interface events which the first node is configured to receive, wherein the portion of graphical code comprises one or more nodes for responding to the one or more user interface events with which the portion of graphical code is associated (attaching program code specifying how the button will respond to a specific user interface event, to the button) (column 10, line 50-column 11, line 29).

Referring to claim 76, Cain teaches wherein the receiving third user input explicitly specifying the one or more user interface events to configure for the first node comprises receiving user input explicitly specifying a first user interface event and a second user interface event (each node, or button can have a plurality of associated events such that a plurality of program codes corresponding to the plurality of user-defined methods are executed in response to the plurality of events) (column 11, line 55-column 12, line 63 and Figures 6E and 6F); wherein the associating the one or more portions of graphical code with the first node comprises associating a first portion of graphical code with the first node, wherein the first portion of graphical code comprises a plurality of interconnected nodes visually indicating functionality for responding to the first user interface event; wherein the associating the one or more portions of graphical code with the first node further comprises associating a second portion of graphical code with the first node, wherein the second portion of graphical code comprises a plurality of interconnected nodes visually indicating functionality for responding to the second user interface event (each node, or button can have a plurality of associated events such that a plurality of program codes corresponding to the plurality of user-defined methods are executed in response to the plurality of events; furthermore, the plurality of interconnected nodes shown in Figure 5D visually indicate functionality such as containership relationships which controls how user interface events are handled) (column 11, line 55-column 12, line 63, column 14, line 56-column 17, line 67 and Figures 6E and 6F).

Referring to claim 77, Cain teaches executing the graphical program (column 11, lines 17-29); executing the first portion of graphical code associated with the first node in response to the first user interface event being generated during execution of the graphical program; and

executing the second portion of graphical code associated with the first node in response to the second user interface event being generated during execution of the graphical program (each node, or button can have a plurality of associated events such that a plurality of program codes corresponding to the plurality of user-defined methods are executed in response to the plurality of events) (column 11, line 55-column 12, line 63 and Figures 6E and 6F).

Referring to claim 79, Cain teaches wherein the receiving the third user input explicitly specifying the one or more user interface events to configure for the first node comprises graphically connecting a plurality of objects to the first node in response to user input, wherein each object directly represents a user interface event (plurality of objects are connected to other objects, as shown in Figure 5D, each object has an associated user interface events, such as built-in events) (column 11, lines 55-63 and column 14, line 30-column 17, line 24).

Referring to claim 82, Cain teaches creating the portions of graphical code in response to user input (creating code associated with method in response to users creating methods) (column 10, lines 50-58 and column 12, lines 7-29), wherein for each portion of graphical code, creating the portion of graphical code comprises including one or more nodes in the portion of graphical code in response to user input, wherein the one or more nodes are operable to respond to one or more of the user interface events which the first node is configured to receive (the nodes or buttons will respond to user interface events such as clicking the button according to the associated program code that executes with the button is clicked) (column 10, line 50-column 12, line 46).

Referring to claim 87, Cain teaches a method comprising creating a first portion of graphical code in response to user input (creating customized buttons with associated code for

responding to specific user interface events) (column 10, line 50-column 12, line 46 and Figure 4E), wherein the first portion of graphical code comprises one or more nodes that visually indicate functionality for responding to programmatic events generated during execution of the graphical program (the plurality of interconnected nodes visually indicate functionality such as containership relationships which controls how user interface events are handled) (column 14, line 56-column 17, line 67); and configuring the graphical program to dynamically register a first programmatic event during execution of the graphical program (Figure 4E shows the receiving of user input specifying methods corresponding to user interface events to be executed during the run-mode) (column 10, line 50-column 12, line 46), wherein dynamically registering the first programmatic event comprises dynamically associating the first programmatic event with the first portion of graphical code (program code corresponding to how a node, i.e. button respond to specified user interface events are attached to the button so that it executes when the button is selected in the run mode) (column 10, line 50-column 12, line 46); wherein the dynamically registering the first programmatic event causes the first portion of graphical code to execute in response to the first programmatic event being generated (attaching program code to the button so that when the first programmatic event of the button being clicked is received, the code executes) (column 10, line 50-column 11, line 3).

Referring to claim 88, Cain teaches receiving user input specifying the first programmatic event to be dynamically registered during execution of the graphical program (Figure 4E shows the receiving of user input specifying methods corresponding to user interface events to be executed during the run-mode) (column 10, line 50-column 12, line 46); wherein the graphical program is configured to dynamically register the first programmatic event during execution of

the graphical program in response to the user input specifying the first programmatic event (program code corresponding to how a node, i.e. button respond to specified user interface events are attached to the button so that it executes when the button is selected in the run mode) (column 10, line 50-column 12, line 46).

Referring to claim 89, Cain teaches wherein the configuring the graphical program to dynamically register the first programmatic event during execution of the graphical program comprises configuring a node in the graphical program to dynamically register the first programmatic event during execution of the graphical program (program code corresponding to how a node, i.e. button respond to specified user interface events are attached to the button so that it executes when the button is selected in the run mode) (column 10, line 50-column 12, line 46).

Claims 8-9 are rejected under 35 U.S.C. 103(a) as being unpatentable Cain et al. U.S. Patent 5,651,108 (hereinafter "Cain"), as applied to claims 1 and 6 above, and Zizzo U.S. Patent 6,578,174.

Referring to claims 8 and 9, Cain teaches all of the limitations as applied to the claims 1 and 6 above. However, Cain fails to explicitly teach a block diagram executing on a first reconfigurable instrument and the graphical user interface displayed on a display of a second, connected computer system. Zizzo teaches a method for providing a graphical user interface for creating a graphical program (providing tools for designing circuits) (Zizzo: column 18, lines 36-38) similar to that of Cain. In addition, Zizzo further teaches the design executes on a first reconfigurable instrument (server computer system) and the graphical user interface is displayed

on a display of a second computer system (the circuit design executes on a central server, or first computer system while a plurality of user, or second computer systems, connected to the server through a network displays the user interface used in designing the circuit diagram) (Zizzo: column 4, lines 50-60). It would have been obvious to one of ordinary skill in the art, having the teachings of Cain and Zizzo before him at the time the invention was made, to modify system for executing and displaying block diagrams of Cain to include the use of a client/server network system in executing and designing diagrams, taught by Zizzo. One would have been motivated to make such a combination in order to allow design tools to be readily available and easily used on a variety of computing platforms and operating systems.

(10) Response to Argument

Claims 1, 19, 32

The applicant argues that Cain does not teach the limitation "receiving third user input explicitly specifying one or more user interface events to configure for the first node" because Cain teaches configuring methods for the first node and methods are not the same as events. The examiner respectfully disagrees. According to Cain's definition of an "event" and a "method" in the Glossary in column 5, lines 40-41 and column 6, lines 7-8, an event triggers a method and a method defines an object's (i.e. a node's) response to an events. Therefore, events and methods are mutually inclusive; in other words, since methods and events are paired with each other, changes to an event changes a method and vise versa. Cain teaches that users can configure, i.e. customize the methods attached to events and change the properties of the nodes, i.e. buttons

(column 10, line 50-column 12, line 29). Since the events and methods are mutually inclusive entities that are correlated with each other, changing the method (changing the response to an event), changes the property of the overall event response; therefore, the examiner respectfully argues that Cain's teaching of configuring properties and methods for nodes (button) configures the overall user interface event.

Furthermore, the applicant argues that Cain teaches text-based code as opposed to graphical code which comprises one or more nodes. The examiner respectfully disagrees. As shown in Figure 4I of Cain, the first node (the "Label" button) is associated with graphical code, such as the graphically displayed code 491, as recited in column 13, lines 6-17; box 491 is a graphical box showing the visual/hierarchical relationship of the button and not text. Furthermore, box 510 in Figure 5D shows graphical code comprising one or more nodes associated with a node; for example, the first node is linked to each of the other nodes shown in the Object Tree in a graphical/hierarchical relationship. This is further recited in column 16, line 32-column 18, line 41 and further shown in Figure 6B for example. Therefore, the examiner respectfully argues that Cain teaches "associating one or more portions of graphical code with the first node in response to fourth user input, wherein each portion of graphical code comprises one or more nodes for responding to one or more of the user interface events which the first node is configured to receive".

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claims 2, 20, 33

The applicant argues that Cain's button does not comprise one or more sub-diagrams, and Cain does not teach displaying portions of graphical code within sub-diagrams of the button, as discussed with reference to claim 1. The examiner respectfully disagrees. As shown in Figures 5A-H and 6A-M, the Object Tree graphically shows the hierarchical relationship between nodes. For example, as shown in Figure 6B, the "#Page2" node contains sub-nodes "#Box3", "#Box6" and "#Text8"; in other words, nodes "#Box3", "#Box6" and "#Text8" are sub-diagrams graphically displayed within the "#Page2" node.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claims 3, 21, 34

The applicant argues that Cain does not teach sub-diagrams as noted with reference to claim 2 and events as noted with reference to claim 1. The examiner refers to the response to the arguments with reference to claims 1 and 2 above and maintains that Cain teaches sub-diagrams and events. Furthermore, the applicant states that Cain does not teach "receiving user input explicitly specifying one or more user interface events to which each of the sub-diagrams of the first node corresponds". The examiner respectfully disagrees. Cain teaches that objects can be contained within other objects and each object displayed on the GUI can have user interface events specified by the user, which causes a method to execute in response to an event such as user selection, as recited in column 10, line 48-column 11, line 3 and column 14, line 56-column 17, line 24.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claims 4, 35

The applicant argues that Cain does not teach graphical code, but teaches text-based code instead. The examiner respectfully disagrees. As shown in Figure 4I of Cain, the first node (the "Label" button) is associated with graphical code, such as the graphically displayed code 491, as recited in column 13, lines 6-17; box 491 is a graphical box showing the visual/hierarchical relationship of the button and not text. Furthermore, box 510 in Figure 5D shows graphical code comprising one or more nodes associated with a node; for example, the first node is linked to each of the other nodes shown in the Object Tree in a graphical/hierarchical relationship. This is further recited in column 16, line 32-column 18, line 41 and further shown in Figure 6B for example. Therefore, the examiner respectfully argues that Cain teaches "associating one or more portions of graphical code with the first node in response to fourth user input, wherein each portion of graphical code comprises one or more nodes for responding to one or more of the user interface events which the first node is configured to receive".

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claims 6, 7

The applicant argues that Cain does not teach graphical code, but teaches text-based code instead. The examiner respectfully disagrees. As shown in Figure 4I of Cain, the first node (the

"Label" button) is associated with graphical code, such as the graphically displayed code 491, as recited in column 13, lines 6-17; box 491 is a graphical box showing the visual/hierarchical relationship of the button and not text. Furthermore, box 510 in Figure 5D shows graphical code comprising one or more nodes associated with a node; for example, the first node is linked to each of the other nodes shown in the Object Tree in a graphical/hierarchical relationship. This is further recited in column 16, line 32-column 18, line 41 and further shown in Figure 6B for example.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 12

The applicant argues that Cain does not teach displaying graphical code associated with the first node within the first node. The examiner respectfully disagrees. As shown in Figure 4I of Cain, the first node (the "Label" button) is associated with graphical code, such as the graphically displayed code 491, as recited in column 13, lines 6-17; box 491 is a graphical box showing the visual/hierarchical relationship of the button and not text. Also, box 510 in Figure 5D shows graphical code comprising one or more nodes associated with a node; for example, the first node is linked to each of the other nodes shown in the Object Tree in a graphical/hierarchical relationship. As further shown in Figures 5A-H and 6A-M, the Object Tree graphically shows the hierarchical relationship between nodes. For example, as shown in Figure 6B, the "#Page2" node contains sub-nodes "#Box3", "#Box6" and "#Text8"; in other words, nodes "#Box3", "#Box6" and "#Text8" are sub-diagrams graphically displayed *within* the "#Page2" node.

Column 16, lines 53-67 also recited embedding objects *within* another object.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claims 13 and 14

The applicant argues that Cain does not teach user interface events, as noted with respect to claim 1. The examiner refers to the response to the arguments with reference to claim 1 above and maintain that Cain teaches configuring user interface events. The applicant further argues that the push-button events specify the action performed on the push-button but do not explicitly specify the push-button itself. The examiner respectfully disagrees. The user interface event of the user clicking the button shown in Figure 4I results in the display of a user interface element of the "Greetings" box; therefore, a specific action such as the display of a box is specified for a particular user interface element such as the button; without a user interface element such as the button, there can be no action specified because the action would not be correlated with an object; therefore, the action of displaying the "Greetings" box upon user selection of the button explicitly specifies the button and the action.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 15

The applicant argues that Cain teaches methods instead of user interface events, as noted with respect to claim 1. The examiner refers to the response to the arguments with reference to claim 1 above and maintain that Cain teaches configuring user interface events.

Claims 16, 71

The applicant argues that Cain does not teach dynamically registering a first user interface event during execution of the graphical program because Cain teaches the user interacting with the graphical user interface while editing the graphical program, not during execution of the graphical program. The examiner respectfully disagrees. Cain teaches that the screen can be changed from design mode to run mode, as recited in column 11, lines 14-29. In the run mode, the program is dynamically executed, and Cain explicitly states that "in the run mode, the button can be clicked" (column 11, lines 20-21); therefore, Cain teaches the user interacting with the graphical user interface *during* execution of the graphical program, i.e. in run mode.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 17

The applicant argues that Cain does not teach dynamically registering user interface events during execution of the graphical program, as noted with respect to claim 16. The examiner refers to the response to the arguments with reference to claim 16 above and maintain that Cain teaches dynamically registering user interface events during execution of the graphical program.

Cain teaches that the screen can be changed from design mode to run mode, as recited in column 11, lines 14-29. In the run mode, the program is dynamically executed, and Cain explicitly states that "in the run mode, the button can be clicked" (column 11, lines 20-21); therefore, Cain teaches the user interacting with the graphical user interface during execution of the graphical program, i.e. in run mode.

Furthermore, the applicant states that Cain also does not teach the additional limitation that the second node is configured to dynamically register the first user interface event during execution of the graphical program by connecting the second node to the first node in response to user input. The examiner respectfully disagrees. Cain teaches that user input such as selection and creation of objects and buttons creates the interconnected nodes shown in the "Object Tree", such as seen in Figure 6I.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 18

The applicant argues that Cain teaches the user interacting with the graphical user interface while editing the graphical program, not during execution of the graphical program, as noted with respect to claim 16. The examiner refers to the response to the arguments with reference to claim 16 above. Cain teaches that the screen can be changed from design mode to run mode, as recited in column 11, lines 14-29. In the run mode, the program is dynamically executed, and Cain explicitly states that "in the run mode, the button can be clicked" (column 11, lines 20-21);

therefore, Cain teaches the user interacting with the graphical user interface during execution of the graphical program, i.e. in run mode. Furthermore, Cain teaches that allowing users to un-register, i.e. delete user interface event-method pairings so that they are not part of the program during run-time, as shown by the "Delete" option 466 in Figure 4E.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 69

The applicant argues that Cain does not teach displaying graphical code within sub-diagrams of the button. The examiner respectfully disagrees. The examiner respectfully refers to the response to arguments for claims 1 and 2 above. As shown in Figures 5A-H and 6A-M, the Object Tree graphically shows the hierarchical relationship between nodes. For example, as shown in Figure 6B, the "#Page2" node contains sub-nodes "#Box3", "#Box6" and "#Text8"; in other words, nodes "#Box3", "#Box6" and "#Text8" are sub-diagrams graphically displayed within the "#Page2" node.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 70

The applicant argues that Cain does not teach displaying graphical code associated with the first node within the first node. The examiner respectfully disagrees and refers to the response to arguments for claim 12 above. As shown in Figure 4I of Cain, the first node (the "Label" button) is associated with graphical code, such as the graphically displayed code 491, as recited in

column 13, lines 6-17; box 491 is a graphical box showing the visual/hierarchical relationship of the button and not text. Also, box 510 in Figure 5D shows graphical code comprising one or more nodes associated with a node; for example, the first node is linked to each of the other nodes shown in the Object Tree in a graphical/hierarchical relationship. As further shown in Figures 5A-H and 6A-M, the Object Tree graphically shows the hierarchical relationship between nodes. For example, as shown in Figure 6B, the "#Page2" node contains sub-nodes "#Box3", "#Box6" and "#Text8"; in other words, nodes "#Box3", "#Box6" and "#Text8" are sub-diagrams graphically displayed *within* the "#Page2" node.

Column 16, lines 53-67 also recited embedding objects *within* another object.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 72

The applicant argues that Cain does not receiving third user input explicitly specifying the one or more user interface events to configure for the first node, as noted with reference to claim 1. The examiner refers to the response to the arguments with reference to claim 1 above and maintain that Cain teaches receiving third user input explicitly specifying the one or more user interface events to configure for the first node.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 73

The applicant argues that Cain does not teach graphical code, as noted above. The examiner respectfully refers to the response to arguments for claims 1-3 above.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claims 76, 77

The applicant argues that Cain teaches text-based code, and not graphical code. The examiner respectfully disagrees. The examiner respectfully refers to the response to arguments for claim 1 above. As shown in Figure 4I of Cain, the first node (the "Label" button) is associated with graphical code, such as the graphically displayed code 491, as recited in column 13, lines 6-17; box 491 is a graphical box showing the visual/hierarchical relationship of the button and not text. Furthermore, box 510 in Figure 5D shows graphical code comprising one or more nodes associated with a node; for example, the first node is linked to each of the other nodes shown in the Object Tree in a graphical/hierarchical relationship. This is further recited in column 16, line 32-column 18, line 41 and further shown in Figure 6B for example. Therefore, the examiner respectfully argues that Cain teaches "associating one or more portions of graphical code with the first node in response to fourth user input, wherein each portion of graphical code comprises one or more nodes for responding to one or more of the user interface events which the first node is configured to receive".

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 79

The applicant argues that Cain does not teach receiving user input explicitly specifying one or more user interface events by graphically connecting a plurality of objects to the button, as noted with respect to claim 1. The examiner respectfully refers to the response to arguments for claim 1 above.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 82

The applicant argues that Cain teaches text-based code, and not graphical code. The examiner respectfully disagrees. The examiner respectfully refers to the response to arguments for claim 1 above. As shown in Figure 4I of Cain, the first node (the "Label" button) is associated with graphical code, such as the graphically displayed code 491, as recited in column 13, lines 6-17; box 491 is a graphical box showing the visual/hierarchical relationship of the button and not text. Furthermore, box 510 in Figure 5D shows graphical code comprising one or more nodes associated with a node; for example, the first node is linked to each of the other nodes shown in the Object Tree in a graphical/hierarchical relationship. This is further recited in column 16, line 32-column 18, line 41 and further shown in Figure 6B for example. Therefore, the examiner respectfully argues that Cain teaches "associating one or more portions of graphical code with the first node in response to fourth user input, wherein each portion of graphical code comprises one or more nodes for responding to one or more of the user interface events which the first node is configured to receive".

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 23

The applicant argues that Cain does not teach user interface events and graphical code, as noted with respect to claim 1. The examiner respectfully refers to the response to arguments for claim 1 above.

The applicant argues that Cain does not teach the limitation "receiving user input explicitly specifying one or more user interface events to configure for the first node" because Cain teaches configuring methods for the first node and methods are not the same as events. The examiner respectfully disagrees. According to Cain's definition of an "event" and a "method" in the Glossary in column 5, lines 40-41 and column 6, lines 7-8, an event triggers a method and a method defines an object's (i.e. a node's) response to an events. Therefore, events and methods are mutually inclusive; in other words, since methods and events are paired with each other, changes to an event changes a method and vise versa. Cain teaches that users can configure, i.e. customize the methods attached to events and change the properties of the nodes, i.e. buttons (column 10, line 50-column 12, line 29). Since the events and methods are mutually inclusive entities that are correlated with each other, changing the method (changing the response to an event), changes the property of the overall event response; therefore, the examiner respectfully argues that Cain's teaching of configuring properties and methods for nodes (button) configures the overall user interface event.

Furthermore, the applicant argues that Cain teaches text-based code as opposed to graphical code, which comprises one or more nodes. The examiner respectfully disagrees. As shown in Figure 4I of Cain, the first node (the "Label" button) is associated with graphical code, such as the graphically displayed code 491, as recited in column 13, lines 6-17; box 491 is a graphical box showing the visual/hierarchical relationship of the button and not text. Furthermore, box 510 in Figure 5D shows graphical code comprising one or more nodes associated with a node; for example, the first node is linked to each of the other nodes shown in the Object Tree in a graphical/hierarchical relationship. This is further recited in column 16, line 32-column 18, line 41 and further shown in Figure 6B for example. Therefore, the examiner respectfully argues that Cain teaches "associating one or more portions of graphical code with the first node in response to fourth user input, wherein each portion of graphical code comprises one or more nodes for responding to one or more of the user interface events which the first node is configured to receive".

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 24

The applicant argues that Cain teaches text-based code that does not comprise a plurality of interconnected nodes that visually indicate the event response functionality. The examiner respectfully disagrees. As shown in Figure 4I of Cain, the first node (the "Label" button) is associated with graphical code, such as the graphically displayed code 491, as recited in column

13, lines 6-17; box 491 is a graphical box showing the visual/hierarchical relationship of the button and not text. Furthermore, box 510 in Figure 5D shows graphical code comprising one or more nodes associated with a node; for example, the first node is linked to each of the other nodes shown in the Object Tree in a graphical/hierarchical relationship. This is further recited in column 16, line 32-column 18, line 41 and further shown in Figure 6B for example. Therefore, the examiner respectfully argues that Cain teaches "associating one or more portions of graphical code with the first node in response to fourth user input, wherein each portion of graphical code comprises one or more nodes for responding to one or more of the user interface events which the first node is configured to receive".

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 25

The applicant argues that Cain teaches text-based code instead of graphical code, as noted with respect to claim 24. The examiner respectfully refers to the response to arguments for claim 24 above.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claims 36, 53, 66, 86

The applicant argues that Cain does not teach user interface events and graphical code, as noted with respect to claim 1. The examiner respectfully refers to the response to arguments for claims 1 and 23 above.

The applicant argues that Cain does not teach the limitation "receiving user input explicitly specifying one or more user interface events to configure for the first node" because Cain teaches configuring methods for the first node and methods are not the same as events. The examiner respectfully disagrees. According to Cain's definition of an "event" and a "method" in the Glossary in column 5, lines 40-41 and column 6, lines 7-8, an event triggers a method and a method defines an object's (i.e. a node's) response to an events. Therefore, events and methods are mutually inclusive; in other words, since methods and events are paired with each other, changes to an event changes a method and vise versa. Cain teaches that users can configure, i.e. customize the methods attached to events and change the properties of the nodes, i.e. buttons (column 10, line 50-column 12, line 29). Since the events and methods are mutually inclusive entities that are correlated with each other, changing the method (changing the response to an event), changes the property of the overall event response; therefore, the examiner respectfully argues that Cain's teaching of configuring properties and methods for nodes (button) configures the overall user interface event.

Furthermore, the applicant argues that Cain teaches text-based code as opposed to graphical code, which comprises one or more nodes. The examiner respectfully disagrees. As shown in Figure 4I of Cain, the first node (the "Label" button) is associated with graphical code, such as

the graphically displayed code 491, as recited in column 13, lines 6-17; box 491 is a graphical box showing the visual/hierarchical relationship of the button and not text. Furthermore, box 510 in Figure 5D shows graphical code comprising one or more nodes associated with a node; for example, the first node is linked to each of the other nodes shown in the Object Tree in a graphical/hierarchical relationship. This is further recited in column 16, line 32-column 18, line 41 and further shown in Figure 6B for example. Therefore, the examiner respectfully argues that Cain teaches "associating one or more portions of graphical code with the first node in response to fourth user input, wherein each portion of graphical code comprises one or more nodes for responding to one or more of the user interface events which the first node is configured to receive".

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claims 38, 55

The applicant argues that Cain teaches text-based code that does not comprise a plurality of interconnected nodes that visually indicate the event response functionality. The examiner respectfully disagrees. As shown in Figure 4I of Cain, the first node (the "Label" button) is associated with graphical code, such as the graphically displayed code 491, as recited in column 13, lines 6-17; box 491 is a graphical box showing the visual/hierarchical relationship of the button and not text. Furthermore, box 510 in Figure 5D shows graphical code comprising one or more nodes associated with a node; for example, the first node is linked to each of the other nodes shown in the Object Tree in a graphical/hierarchical relationship. This is further recited in

column 16, line 32-column 18, line 41 and further shown in Figure 6B for example. Therefore, the examiner respectfully argues that Cain teaches "associating one or more portions of graphical code with the first node in response to fourth user input, wherein each portion of graphical code comprises one or more nodes for responding to one or more of the user interface events which the first node is configured to receive".

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claims 39, 56

The applicant argues that Cain teaches text-based code that does not comprise a plurality of interconnected nodes that visually indicate the event response functionality. The examiner respectfully disagrees. As shown in Figure 4I of Cain, the first node (the "Label" button) is associated with graphical code, such as the graphically displayed code 491, as recited in column 13, lines 6-17; box 491 is a graphical box showing the visual/hierarchical relationship of the button and not text. Furthermore, box 510 in Figure 5D shows graphical code comprising one or more nodes associated with a node; for example, the first node is linked to each of the other nodes shown in the Object Tree in a graphical/hierarchical relationship. This is further recited in column 16, line 32-column 18, line 41 and further shown in Figure 6B for example. Therefore, the examiner respectfully argues that Cain teaches "associating one or more portions of graphical code with the first node in response to fourth user input, wherein each portion of graphical code comprises one or more nodes for responding to one or more of the user interface events which the first node is configured to receive".

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claims 40, 57

The applicant argues that Cain does not teach displaying graphical code associated with the first node within the first node. The examiner respectfully disagrees. As shown in Figure 4I of Cain, the first node (the "Label" button) is associated with graphical code, such as the graphically displayed code 491, as recited in column 13, lines 6-17; box 491 is a graphical box showing the visual/hierarchical relationship of the button and not text. Also, box 510 in Figure 5D shows graphical code comprising one or more nodes associated with a node; for example, the first node is linked to each of the other nodes shown in the Object Tree in a graphical/hierarchical relationship. As further shown in Figures 5A-H and 6A-M, the Object Tree graphically shows the hierarchical relationship between nodes. For example, as shown in Figure 6B, the "#Page2" node contains sub-nodes "#Box3", "#Box6" and "#Text8"; in other words, nodes "#Box3", "#Box6" and "#Text8" are sub-diagrams graphically displayed *within* the "#Page2" node. Column 16, lines 53-67 also recited embedding objects *within* another object.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claims 41, 44, 58, 60, 67, 78, 83, 84

The applicant argues that Cain teaches allowing users to change methods, and not user interface events. The examiner respectfully disagrees. According to Cain's definition of an "event" and a

"method" in the Glossary in column 5, lines 40-41 and column 6, lines 7-8, an event triggers a method and a method defines an object's (i.e. a node's) response to an events. Therefore, events and methods are mutually inclusive; in other words, since methods and events are paired with each other, changes to an event changes a method and vise versa. Cain teaches that users can configure, i.e. customize the methods attached to events and change the properties of the nodes, i.e. buttons (column 10, line 50-column 12, line 29). Since the events and methods are mutually inclusive entities that are correlated with each other, changing the method (changing the response to an event), changes the property of the overall event response; therefore, the examiner respectfully argues that Cain's teaching of configuring properties and methods for nodes (button) configures the overall user interface event.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claims 43, 68, 85

The applicant argues that Cain does not teach receiving user input that explicitly specifies a name of a first user interface event to associate with the button because Cain teaches methods and not events. The examiner respectfully disagrees. Cain teaches that users can create new and custom methods (the methods are associated with a name) via the interface shown in Figure 4E. According to Cain's definition of an "event" and a "method" in the Glossary in column 5, lines 40-41 and column 6, lines 7-8, an event triggers a method and a method defines an object's (i.e. a node's) response to an events. Therefore, events and methods are mutually inclusive; in other words, since methods and events are paired with each other, changes to an event changes a

method and vise versa. Cain teaches that users can configure, i.e. customize the methods attached to events and change the properties of the nodes, i.e. buttons (column 10, line 50-column 12, line 29). Since the events and methods are mutually inclusive entities that are correlated with each other, changing the method (changing the response to an event), changes the property of the overall event response; therefore, the examiner respectfully argues that Cain's teaching of configuring properties and methods for nodes (button) configures the overall user interface event.

Claims 49, 50

The applicant argues that Cain teaches text-based code as opposed to graphical code, which comprises one or more nodes, as discussed with reference to previous claims. The examiner respectfully disagrees. As shown in Figure 4I of Cain, the first node (the "Label" button) is associated with graphical code, such as the graphically displayed code 491, as recited in column 13, lines 6-17; box 491 is a graphical box showing the visual/hierarchical relationship of the button and not text. Furthermore, box 510 in Figure 5D shows graphical code comprising one or more nodes associated with a node; for example, the first node is linked to each of the other nodes shown in the Object Tree in a graphical/hierarchical relationship. This is further recited in column 16, line 32-column 18, line 41 and further shown in Figure 6B for example. Therefore, the examiner respectfully argues that Cain teaches "associating one or more portions of graphical code with the first node in response to fourth user input, wherein each portion of graphical code comprises one or more nodes for responding to one or more of the user interface events which the first node is configured to receive".

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 87

The applicant argues that Cain does not teach dynamically registering the first programmatic event during execution of the graphical program. The examiner respectfully disagrees. Cain teaches that the screen can be changed from design mode to run mode, as recited in column 11, lines 14-29. In the run mode, the program is dynamically executed, and Cain explicitly states that "in the run mode, the button can be clicked" (column 11, lines 20-21); therefore, Cain teaches the user interacting with the graphical user interface during execution of the graphical program, i.e. in run mode.

Furthermore, the applicant argues that Cain teaches text-based code instead of graphical code. The examiner respectfully disagrees. As shown in Figure 4I of Cain, the first node (the "Label" button) is associated with graphical code, such as the graphically displayed code 491, as recited in column 13, lines 6-17; box 491 is a graphical box showing the visual/hierarchical relationship of the button and not text. Furthermore, box 510 in Figure 5D shows graphical code comprising one or more nodes associated with a node; for example, the first node is linked to each of the other nodes shown in the Object Tree in a graphical/hierarchical relationship. This is further recited in column 16, line 32-column 18, line 41 and further shown in Figure 6B for example. Therefore, the examiner respectfully argues that Cain teaches "associating one or more portions of graphical code with the first node in response to fourth user input, wherein each portion of

graphical code comprises one or more nodes for responding to one or more of the user interface events which the first node is configured to receive".

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 88

The applicant argues that Cain does not teach dynamically registering an event, as noted with respect to previous claims. The examiner respectfully refers to the response to arguments for claim 87 above.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Claim 89

The applicant argues that Cain does not teach dynamically registering an event during execution of a graphical program, as noted with respect to previous claims. The examiner respectfully refers to arguments for claim 87 above. Furthermore, the applicant argues that Cain also does not teach that the event is dynamically registered by a node in the graphical program. The examiner respectfully disagrees. Cain teaches that the screen can be changed from design mode to run mode, as recited in column 11, lines 14-29. In the run mode, the program is dynamically executed, and Cain explicitly states, "in the run mode, the button can be clicked" (column 11, lines 20-21); therefore, Cain teaches the user dynamically interacting with a node, i.e. button, during execution of the graphical program, i.e. in run mode.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Section 103 Rejections

With respect to claims 8 and 9, the applicant argues that Zizzo does not teach the concept of executing a graphical program, wherein during execution of the graphical program, a block diagram of the graphical program executes on a first computer system and a graphical user interface of the graphical program is displayed on a display of a second computer system. The examiner respectfully disagrees. Zizzo teaches a graphical user interface that provides tools for designing circuits, as recited in column 18, lines 36-38. Furthermore, Zizzo teaches that the circuit design executes on a server computer and the graphical user interface is displayed on client/user computers, as recited in column 4, lines 50-60; therefore, Zizzo teaches one computer for executing the circuit design and a second computer for displaying the graphical user interface of the circuit design.

The examiner thus submits that Cain teaches the limitations recited in these claims.

Section 112 Rejections

The applicant's arguments with regards to claims 16, 71 and 87 are moot in view of the withdrawal of the 35 U.S.C. 112 first paragraph rejection.

With respect to claim 80, the applicant argues that the specification provides clear basis for the limitation of "wherein said receiving third user input explicitly specifying the one or more user interface events to configure for the first node does not include receiving user input specifying a connection between the first node and a second node", on page 28, line 26 –page 30, line 12 of the specification. However, although the passages of the specification pointed out by the applicant describes allowing users to specify user interface events for the first node via input to a graphical user interface such as a dialog box, the cited passages do not state that the user is not allowed to specify a connection between the first node and a second node. The MPEP explicitly states "Any negative limitation or exclusionary proviso must have basis in the original disclosure. If alternative elements are *positively recited* in the specification, they may be explicitly excluded in the claims....The mere absence of a positive recitation is not basis for an exclusion" (MPEP 2173.05(i)). Therefore, for claim 80 to properly exclude the alternative element of receiving user input specifying a connection between the first node and a second node, the specification must positively recite this exclusion. Even assuming arguendo that the specification provides basis for the user interface event for the first node being specified by user input to a dialog box without *requiring* the user to specify a connection between the first node and a second node, this basis does not equate to a basis for *excluding* or *not including* the user from specifying a connection between the first node and a second as a means to specify the user interface event. The examiner thus respectfully submits that the applicant's specification does not contain a description of the subject matter claimed in claim 80 in such a way as to reasonably

convey to one skilled in the relevant art that the inventor, at the time the application was filed, had possession of the claimed invention.

With respect to claim 81, the applicant states that similar to the discussion with respect to claim 80, the specification describes an embodiment in which the user may interact with an event configuration dialog to configure a set of events for an event structure node, where the events are configured for the event structure node using the event configuration dialog, independently of configuring other nodes in the block diagram of the graphical program. However, although the passages of the specification pointed out by the applicant describes allowing users to specify user interface events for the first node via input to a graphical user interface such as a dialog box, the cited passages do not mention anything about the specifying being performed independently of configuring other nodes. The MPEP explicitly states "Any negative limitation or exclusionary proviso must have basis in the original disclosure. If alternative elements are *positively recited* in the specification, they may be explicitly excluded in the claims....The mere absence of a positive recitation is not basis for an exclusion" (MPEP 2173.05(i)). Therefore, for the exclusionary proviso of claim 81 to be properly recited, the specification must positively recite that the specifying of the user interface events are performed independently from configuring other nodes in the block. Even assuming arguendo that the specification provides basis for the user interface event for the first node being specified by user input to a dialog box, this basis does not equate to a basis for requiring that the specifying of the user interface events be done independently of configuring other nodes. The examiner thus respectfully submits that the applicant's specification does not contain a description of the subject matter claimed in claim 81

in such a way as to reasonably convey to one skilled in the relevant art that the inventor, at the time the application was filed, had possession of the claimed invention.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

Ting Zhou

Kristine Kincaid
KRISTINE KINCAID
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

Conferees:

Weilun Lo

(Supervisory Patent Examiner)

WEILUN LO
SUPERVISORY PATENT EXAMINER

Kieu Vu

(Primary Examiner)